h e g

Haute école de gestion
Genève

Développement d'une application de fitness

Travail de Bachelor réalisé en vue de l'obtention du Bachelor HES

Par :

Vincent Jalley

Conseiller au travail de Bachelor :

David Billard, Professeur HES

Genève, le 15 Septembre 2017

Haute École de Gestion de Genève (HEG-GE)

Filière Informatique de Gestion

Hes·so GENÈVE
Haute Ecole Spécialisée
de Suisse occidentale

Déclaration

Ce travail de Bachelor est réalisé dans le cadre de l'examen final de la Haute école de gestion de Genève, en vue de l'obtention du titre Bachelor of Science HES-SO en Informatique de Gestion.

L'étudiant atteste que son travail a été vérifié par un logiciel de détection de plagiat.

L'étudiant accepte, le cas échéant, la clause de confidentialité. L'utilisation des conclusions et recommandations formulées dans le travail de Bachelor, sans préjuger de leur valeur, n'engage ni la responsabilité de l'auteur, ni celle du conseiller au travail de Bachelor, du juré et de la HEG.

« J'atteste avoir réalisé seul le présent travail, sans avoir utilisé des sources autres que celles citées dans la bibliographie. »

Fait à Genève, le 15 Septembre 2017

Vincent Jalley

Remerciements

Je remercie particulièrement mon directeur de mémoire, Monsieur David Billard, de m'avoir suivi et d'avoir été aussi disponible que possible pour répondre aux différentes interrogations que je me faisais.

Je remercie par la même occasion, mon collègue de classe, Monsieur Marques Da Silva Joel, avec lequel on a partagé une certaine motivation commune de travail afin d'avancer nos travaux de Bachelor respectifs. En effet, il n'est pas toujours facile de se mettre à travailler durant l'été pendant les vacances et surtout quand on a été étudiant toute sa vie.

Je remercie également, un autre collègue de classe, Monsieur Xavie Paez, d'avoir testé mon application et de m'avoir trouvé les bugs rencontrés pendant l'utilisation.

Je souhaite aussi remercier tous mes proches qui ont su me soutenir dans l'accomplissement de ce projet de fin d'étude.

Pour terminer, je voudrais particulièrement remercier la Haute Ecole de Gestion de Genève et tous les professeurs qui ont participé pour tout ce dont ils m'ont appris durant ces 3 années d'enseignement dans son établissement.

Résumé

La création d'une application Android et IOS peut prendre du temps, surtout si l'on doit dédoubler le code pour les faire fonctionner sur ces mêmes plateformes. Heureusement, il existe des Frameworks permettant de construire des applications IOS, Android, et Windows Phone au moyen d'un seul code source. Dans ce travail de Bachelor, il sera question du développement d'une application de fitness au moyen du Framework Ionic 3 pour produire une application mobile hybride. Celui-ci embarque avec lui le célèbre Framework Angular 2 de Google et les langages Html 5 et CSS 3.

Les compétences acquises à la Haute Ecole de Gestion de Genève sont aux rendez-vous, puisqu'on passera d'un cahier des charges de fonctionnalité à une application complète et fonctionnelle. Si la base de données, le développement JavaScript et la méthode de gestion de projet SCRUM vous intéresse, je vous conseille clairement de poursuivre la lecture de ce travail de Bachelor.

Table des matières

Liste des tableaux

Liste des figures

1. Introduction

Le travail de Bachelor est le projet qui doit conclure ma formation Bachelor à la Haute Ecole de Gestion de Genève. C'est pourquoi, j'ai décidé de me lancer dans la réalisation d'une application mobile, afin de montrer les compétences acquises durant ces trois dernières années.

Le sujet de l'application m'est venu relativement vite. En effet, je pratique le fitness depuis plusieurs années et j'ai toujours eu l'habitude de noter, dans une application de prise de notes, les exercices effectués lors de mes entrainements. N'ayant pas trouvé d'application optimal à ce que je recherchais pour noter mes performances, je me suis dit, pourquoi ne pas faire moi-même cette application de gestion durant mon travail de Bachelor.

L'application mobile aura pour avantage d'apporter de la valeur ajoutée aux entrainements enregistrés dans l'application. Comment me direz-vous ? Tout simplement avec la visualisation de statistiques sur plusieurs caractéristiques. De plus, des succès seront à débloquer pour amener les gens à utiliser l'application et se motiver à toujours en faire plus.

La conception d'une telle application va demander la mise en pratiques de plusieurs domaines enseignés à la HEG, d'une grande partie de la recherche pour connaitre les technologies à utiliser et de les prendre en main, de la modélisation pour le stockage des données dans une base de données, de la création d'interface avec de bonnes ergonomies, du développement pour faire fonctionner le tout et pour terminer une méthode de gestion de projet, indispensable à la réalisation de tout projet informatique.

L'application portera le nom de « Sports Training » qui signifie « Entrainements de sport » en français.

2. Méthode de Gestion de projet

Avant de commencer ce beau projet de développement, il m'a fallu choisir une méthode de gestion de projet. J'ai opté pour SCRUM. Il y en a pleins d'autres, mais cette méthode à l'avantage d'obliger ces utilisateurs à noter des stories pour se souvenir des exigences du mandant, ordonnées par la business value la plus importante. De plus, à la fin de chaque sprint, on doit fournir quelque chose de fonctionnel à la fin de celui-ci. Etant mon propre mandant, cela me permettait de m'organiser pour savoir dans quel ordre je devais m'y prendre.

De plus, j'ai séparé mon projet en 4 sprints de deux semaines et ainsi réparti les stories dans les sprints. J'ai pu me rendre compte, après estimation de chaque story, que le projet était trop ambitieux au départ avec le premier cahier des charges que je mettais fixé. C'est pourquoi j'ai dû mettre de côté le site web et me concentrer uniquement sur l'application mobile. De plus, c'est une méthode que je connais bien, puisque je l'ai exercé durant le GREP en dernière année de la HEG.

La durée du travail de Bachelor est de 8 semaines à temps pleins. Ce qui correspond à un total d'heures à 8h par jour, de 340 heures de travail.

Toutes les informations et documents concernant la méthode SCRUM sont en annexes de ce présent document.

3. Fonctionnalités de l'application

3.1 Gestion des entrainements

3.1.1 Entrainements

La fonctionnalité phare de l'application est la gestion des entrainements. Un entrainement est le nom que j'ai choisi pour regrouper une séance d'entrainement contenant plusieurs exercices. Celui-ci a plusieurs caractéristiques comme un nom, une date de réalisation et ainsi qu'une durée de séance approximative.

Il est possible d'afficher une liste d'entrainement, d'afficher les détails d'une séance et ainsi pouvoir visualiser les caractéristiques citées plus haut, mais aussi les exercices contenus dans cet entrainement. L'ajout, la modification et la suppression d'un entrainement sont aussi des fonctionnalités présentes dans l'application.

3.1.2 Exercices

Les exercices sont de type musculation ou cardiovasculaire comme ceux que l'on trouve dans les salles de sport. Une fois un entrainement créé, on a la possibilité d'ajouter un exercice connu de l'application dans une séance d'entrainement. Cette fonctionnalité se veut très simple, une liste d'exercice est affichée et on choisit parmi ceux proposés. Il y a aussi la possibilité de créer un exercice n'étant pas déjà dans l'application.

Un exercice est constitué d'un nom, d'une description, d'un type et du muscle principalement travaillé durant son exécution. De plus, une fois un exercice ajouté à un entrainement, on peut lui rajouter des séries ou enlever l'exercice de l'entrainement.

3.1.3 Séries

Une série est contenue dans un exercice. Elle représente le nombre de fois que l'on va répéter une action, par exemple, 4 séries de 12 répétitions à 20 kilos ou pour du cardio 20 min de vélo. Elles seront exclusivement ajoutées aux exercices concernés et donc chaque série est unique.

3.2 Profil

Dans cette partie, on y retrouvera toutes les informations concernant l'utilisateur de l'application. En effet, il peut être pratique de connaître l'âge d'une personne, ainsi que plusieurs autres mensurations. Le point fort de cette fonctionnalité, c'est que l'utilisateur pourra une fois par jour ajouter une mesure pour une mensuration en particulier. Par exemple, il peut ajouter son poids ou son tour de bras chaque jour et visualiser un historique des mesures. De plus, sur la page d'accueil du profil, il est affiché les dernières données relatives à chaque mensuration.

3.3 Statistiques

Les statistiques sont importantes, car ce sont elles qui feront la différence entre un entrainement écrit dans un bloc note et l'application mobile. Il est question d'afficher par exemple dans des graphiques le nombre d'entrainement regroupé par mois, les poids portés durant les séries d'un exercice en particulier pour voir l'évolution. De plus, On peut afficher des graphiques en lien avec les mensurations enregistrées, mais encore les muscles travaillés durant les exercices.

3.4 Glossaire

Le glossaire est là pour consulter les détails des exercices et des muscles contenus dans l'application. En effet, il peut être intéressant de lire les détails d'un exercice avant

de le pratiquer ou encore d'afficher les exercices par muscle et ainsi découvrir la position du muscle sur le corps humain.

3.5 Succès

Cette fonctionnalité est un élément qui permettra de motiver les utilisateurs à continuer leur progression. Il y a un système avec de l'expérience ou chaque entrainement, exercice, série réalisée fera gagner de l'expérience ainsi débloquer des badges dans l'application.

4. Technologies utilisées

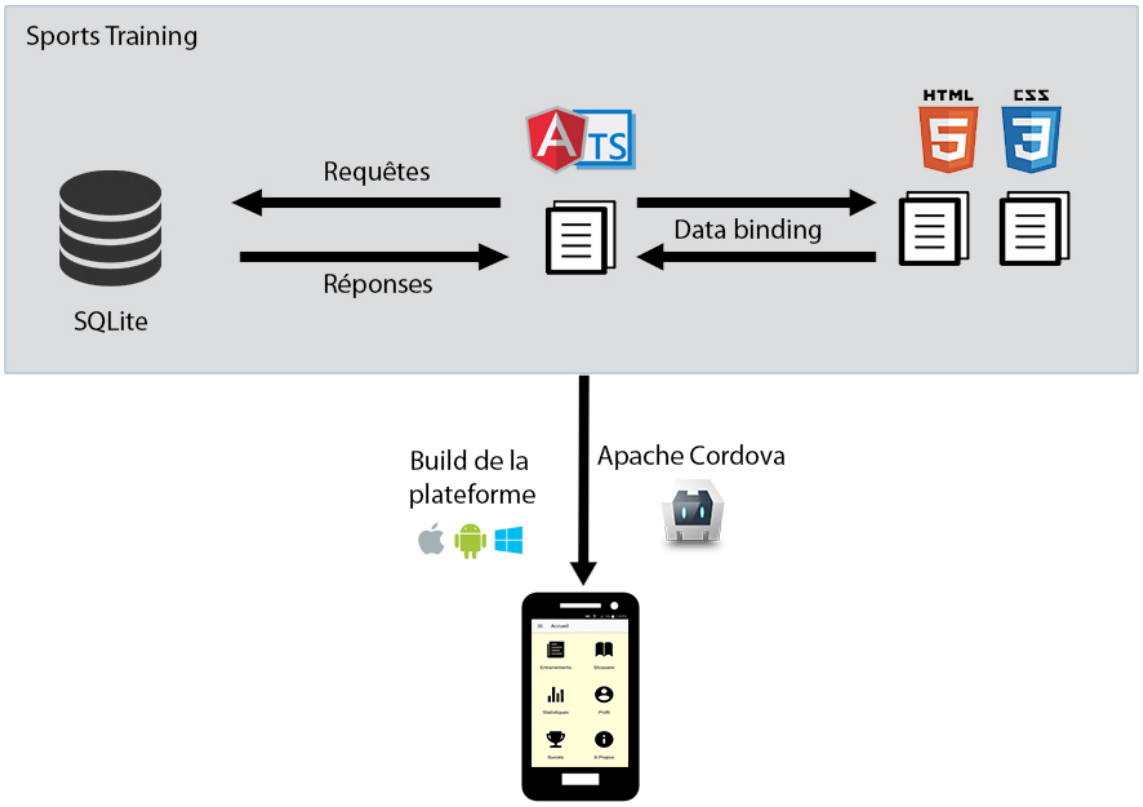
4.1 Framework Ionic

Il y a un bon nombre de technologie qui pourraient convenir à la réalisation d'une application mobile. Ionic n'est pas le premier à faire cela, il existe JqueryMobile, SenchaTouch. Cependant, j'ai choisi l'utilisation du Framework Ionic, car Je connaissais ce Framework de nom et que le groupe avec lequel mon équipe et moi avons collaboré pendant le GREP ont développé leur application avec cette technologie. De plus, je voulais apprendre à utiliser celui-ci pour un futur projet professionnel, car je vais devoir réaliser la maintenance d'une application web développée avec le Framework Ionic.

En effet, ce Framework dispose de plusieurs avantages. Il peut produire des applications mobiles hybrides. C'est-à-dire qu'il y aura besoin d'un seul code source pour créer des versions sous Android, IOS, Windows Phone ou encore BlackBerry. Tout ceci se fera avec l'aide d'Apache Cordova qui permet de publier le projet en une application réelle. Par exemple pour construire l'exécutable sur la plateforme Android, Cordova construira un exécutable avec l'extension « APK ». De plus, la réalisation d'une application hybride combine les éléments HTML 5 d'une application web et d'une application native pour l'utilisation des fonctionnalités natives des smartphones (vibreur, notification, etc..).

Ensuite, Le Framework Ionic est relativement simple à prendre en main. Il a la particularité d'utiliser des technologies avec lequel j'ai eu à travailler par le passé. En effet, toutes les interfaces sont réalisées au moyen des langages HTML 5 et SCSS (dernières versions des langages HTML et CSS), alors que la partie logique est faite avec Angular 2 qui est un Framework JavaScript récent. On peut rajouter que la documentation sur le Framework Ionic est très bien travaillée et structurée. Ce qui fait un très bon point pour son apprentissage et son utilisation.

Figure 1 – Architecture de l'application avec le Framework Ionic



4.2 Framework Angular 2

Ionic embarque avec lui le célèbre Framework Angular2. C'est un Framework JavaScript permettant de développer des applications web. Il possède deux points forts.

Le premier, comme tout Framework, il nous impose de coder à la façon du Framework et celui-ci repose sur le design pattern « Model View ViewMode »l. Le principe est simple, Le modèle représente les données, la vue représente les pages, ainsi que les éléments du DOM et le dernier représente le data-binding permettant d'avoir des données similaires entre le modèle et la vue. Ce qui veut dire que chaque changement dans le modèle impactera les données de la vue immédiatement.

Le second, c'est sa notion de « components » ou anciennement « directive ». Tous les éléments proposés dans la documentation de Ionic sont proposés sous cette forme (d'un contrôleur, d'une vue et d'une feuille de style). De plus, on peut faire appel à n'importe quel composant, grâce à l'injection de dépendance dans le constructeur du contrôleur.

4.3 Stockage des données

Cette partie constitue le backend de mon application, c'est-à-dire la partie en arrière-plan pour l'accès aux données de mon application. J'ai décidé d'opter pour le module SQLite disponible dans Ionic. Ce module est un système de base de données permettant d'utiliser un moteur de bases de données relationnelles pour le stockage des données en natif sur le téléphone. Il a la caractéristique de fonctionner sans une connexion Internet, ce qui est un très bon point étant donné que dans les salles de sport, le réseau n'est vraiment pas terrible.

De plus, si un jour je voulais rajouter un serveur distant disposant de cette même base de données, pour synchroniser les données par exemple, cela peut être réalisé facilement.

Les interactions avec la base de données se font comme sur n'importe quelle base de données relationnelles, au moyen de requêtes (select, update, delete, etc..). À noter que SQLite à sa propre documentation en ligne et que Ionic en a aussi fait une. Comme je l'ai précisé avant, une documentation au top.

4.4 HTML 5

Le Framework Ionic utilise la dernière version du HTML. La version HTML 5 a été finalisée en octobre 2014. Le html est un langage balisé permettant de structurer les données d'une page Internet.

4.5 SCSS

Le Framework Ionic embarque aussi avec lui le SCSS. Ce format de fichier à la même particularité que les fichiers CSS pour formater le texte. Cependant, il dispose de fonctionnalités syntaxiques en plus, comme des variables par exemple. Il est à noter qu'un fichier CSS est un fichier SCSS valide car le SCSS n'est qu'une extension de ce format.

5. Environnement de développement

5.1 Github

Github est l'outil que j'ai utilisé pour gérer les différents codes sources du développement de l'application. En effet, j'ai travaillé sur deux ordinateurs différents (mon ordinateur fixe et mon portable) et ce logiciel m'a permis de garder à jour les différents codes sources peu importe sur quel machine j'étais en train de travailler. De plus, cela me permettait de garder un backup de mes fichiers en cas de perte de données.

5.2 Sublime Text 3

Sublime Text 3 est l'éditeur de texte que j'utilise quand je fais du développement web. Il a un nombre important de fonctionnalités autres que le simple traitement de texte comme par exemple des auto-complétions des fonctions de base de plusieurs langages, des beautifiers, des correcteurs syntaxiques, un gestionnaire de projet, etc. On peut vraiment y installer ce qu'on veut, puisqu'il y a la possibilité d'installer les packages que l'on veut.

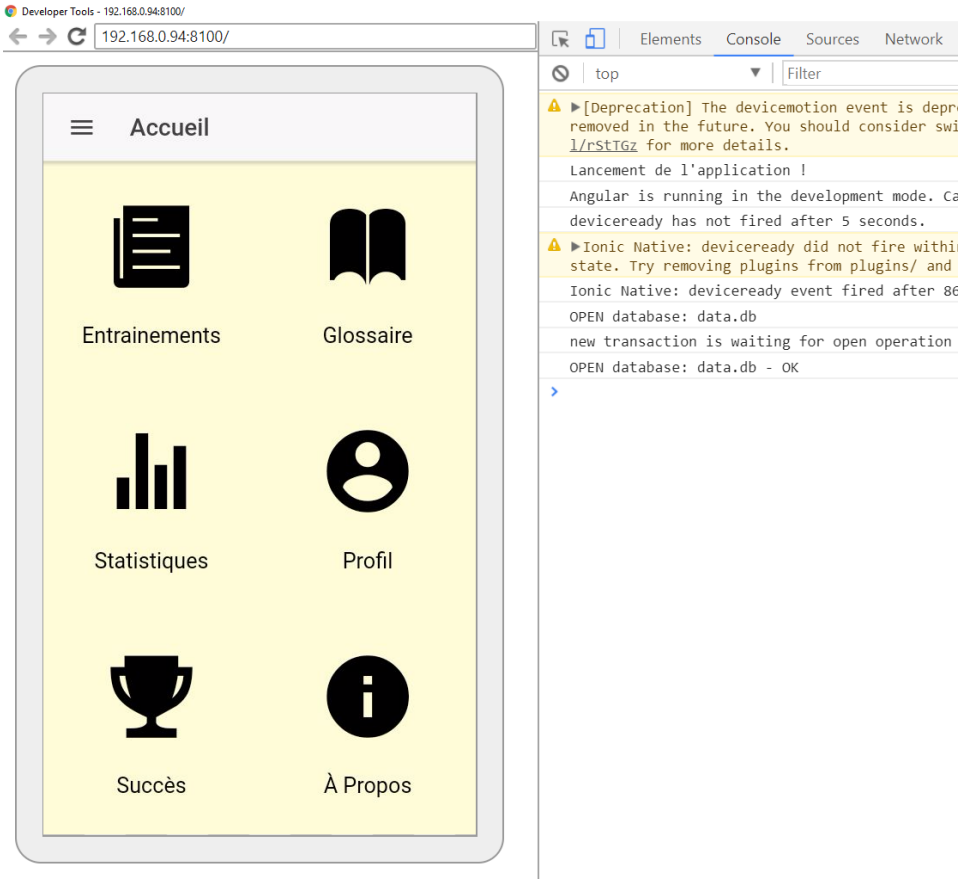
5.3 NodeJs

NodeJs permet de créer un environnement côté serveur qui va permettre d'utiliser le JavaScript pour générer les pages web de l'application. On a en effet besoin de NodeJs pour tester les scripts Angular 2 en mode développement.

5.4 Emulateur, smartphone, navigateur

Pour tester le rendu de l'application, j'ai utilisé dans un premier temps mon navigateur avec le serveur de NodeJs. Cela fonctionnait au début puisque je faisais uniquement de l'interface web. Ensuite, j'ai dû passer par un émulateur Android pour tester les fonctions natives du téléphone que je devais mettre en place. J'ai trouvé un peu laborieux de passer par un émulateur, alors je me suis vite tourné vers le rendu sur un périphérique physique. En effet, mon smartphone sous Android fonctionnait plus rapidement et je pouvais avec une extension de chrome afficher le mode console de l'application et mener des actions sur le téléphone avec la souris.

Figure 2 – Utilitaire du navigateur Google Chrome pour visualiser le smartphone connecté et la console



5.5 DB Browser for SQLite

Ce logiciel m'a été très utile en ce qui concerne les requêtes SQL, puisqu'il me permettait de tester les requêtes avant même de les mettre dans le code de l'application. En effet, SQLite est comme son nom l'indique « léger », donc il y a quelques fonctions qui ne sont pas disponibles. De plus, il n'y a que 4 types différents pour le stockage de données, ce qui fut un peu déroutant au début pour le traitement des dates par exemple.

6. Données

6.1 Description

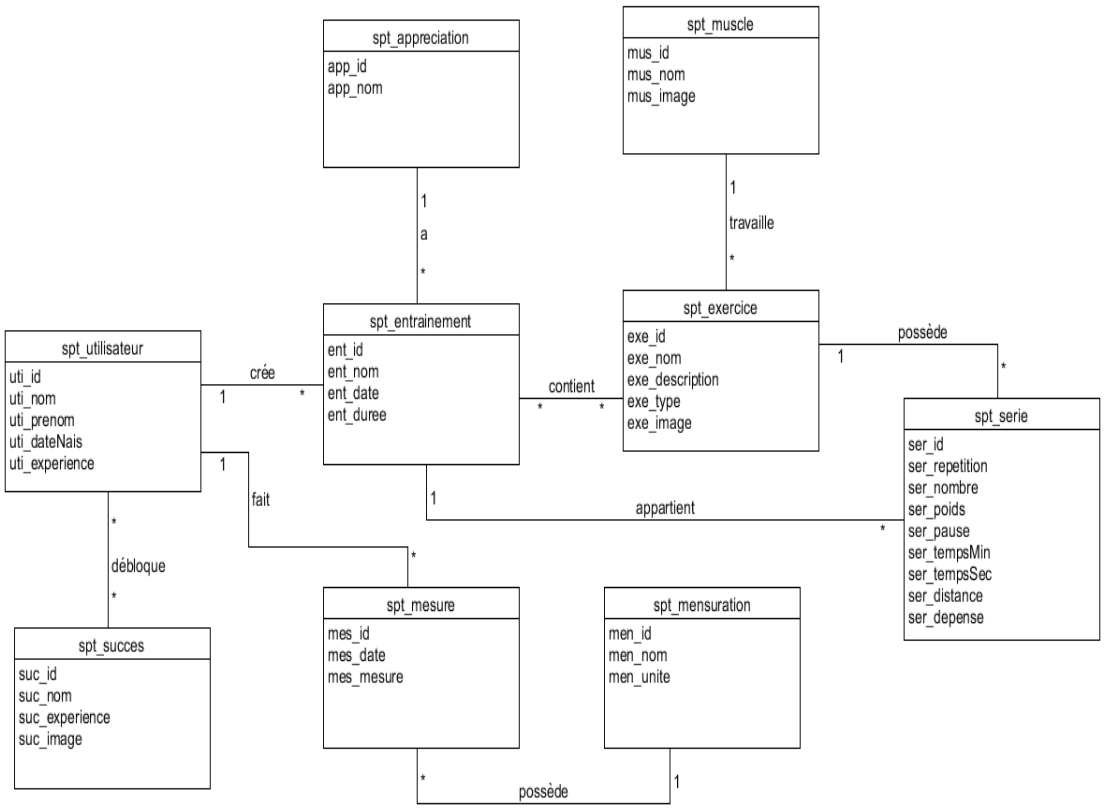
J'ai choisi de m'orienter vers une base de données SQLite avec un modèle de données relationnelles, car pour ce projet, il me fallait une base de données bien structurée pour exécuter les bonnes requêtes. Par conséquent, j'ai pu mettre en pratique les compétences acquises durant la HEG et appliquer les différentes normes notamment pour les noms des tables, attributs, cardinalités, etc…

J'ai commencé par concevoir le premier modèle avec le MCD, afin d'identifier les données qu'il me fallait stocker dans les différentes tables. Ensuite, j'ai réalisé le MLD pour définir les clés étrangères et les nouvelles tables qui devaient être créées dû aux différentes règles de modélisation.

6.2 Modèles de données

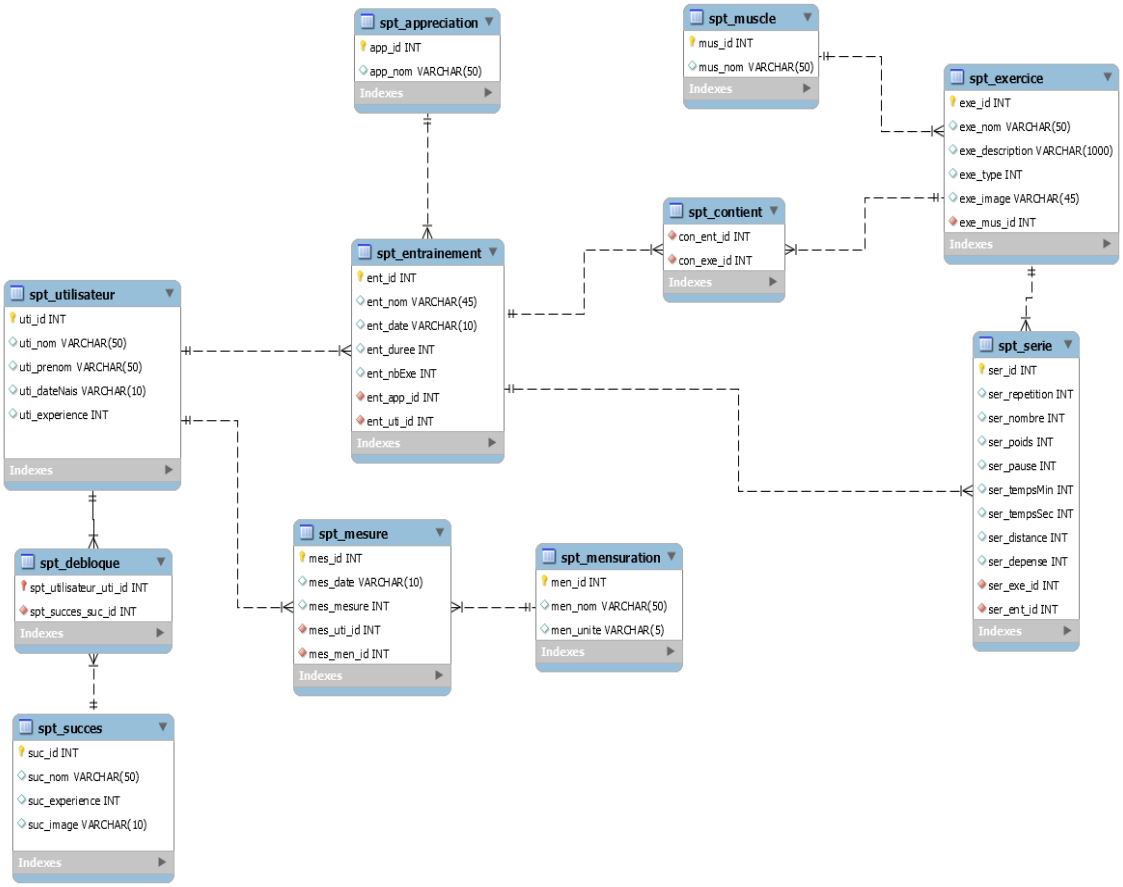
6.2.1 Modèle Conceptuel de Données (MCD)

Figure 3 – Modèle Conceptuel de Données de l'application Sports Training



6.2.2 Modèle Logique de Données (MLD)

Figure 4 - Modèle Logique de Données de l'application Sports Training



6.3 Provenance des données de base

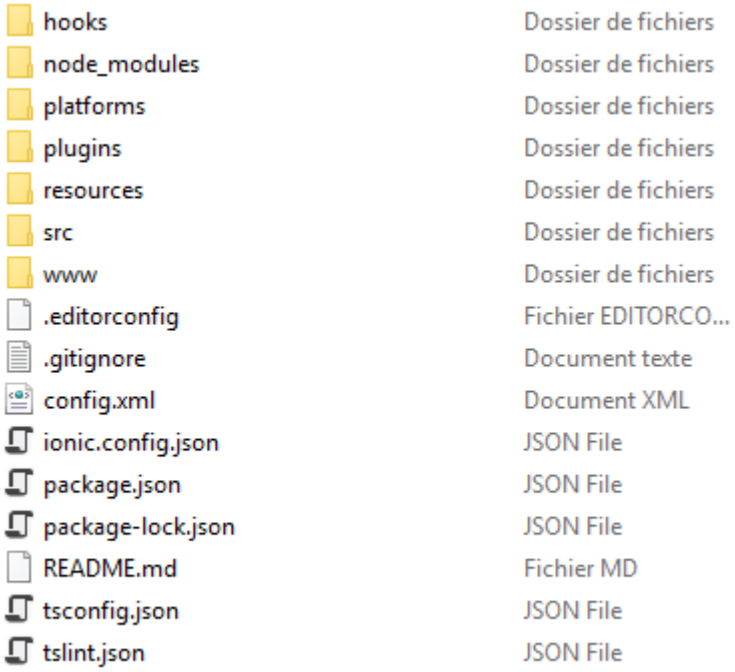
Les données présentes au départ dans l'application proviennent de différentes sources. Pour ce qui est des exercices de fitness contenant des machines de musculation, elles viennent du site web de la marque TechnoGym. Il m'a été utile de récupérer les images, les noms, les descriptions et les muscles travaillés de ces exercices, car je ne dispose pas des compétences nécessaires dans le monde de la musculation pour tout faire moi-même. Ensuite, quelques autres exercices sont tirés d'un site de musculation « Espace-musculation ». Bien entendu, s'il était question de commercialiser l'application, il faudrait trouver un accord pour autoriser l'exploitation de ces données externes. Toutes les autres données sont donc des données imaginées pour l'application par mes soins.

7. Arborescence de l'application

7.1 Dossier complet

Il faut savoir que toutes cette arborescence est créée automatiquement par le Framework Ionic au moment de la création d'un nouveau projet. On peut le remercier, car à première vue cela à l'air bien complexe. Je vais en expliquer les principaux dossiers.

Figure 5 – contenu du répertoire principal du projet Ionic



Le dossier « hooks » sert de stockage pour des scripts afin de personnaliser les commandes de Cordova. Ensuite, le répertoire « node_module » contient tous les fichiers indispensables pour l'utilisation du serveur nodeJs. Le dossier « platforms » est très important. En effet, c'est dans celui-ci qu'on retrouvera les builds produits par Apache Cordova des différentes plateformes. Dans le répertoire « plugins », on y retrouve tous les modules ajoutés par nos soins avec les commandes de cordova pour faire communiquer l'application web avec les parties natives du smartphone, on y retrouvera par exemple le module sqlite-storage. Ensuite dans « resources », on y stockera les images du logo de l'application dans différentes tailles, et ainsi pour différentes plateformes. Après, le dossier « src » qui contient le code source de l'application et pour terminer le répertoire « www » qui contient les pages principales de l'application web comme l'index.html par exemple.

Ensuite, il y quelques fichiers, cependant, on va s'intéresser uniquement au fichier « config.xml », car il contient toutes les informations à propos de l'application, par exemple, le nom, les icones ou encore la description de l'application. C'est via ce fichier là qu'Apache Cordova va prendre les données générales de l'application pour les afficher correctement dans la plateforme voulue.

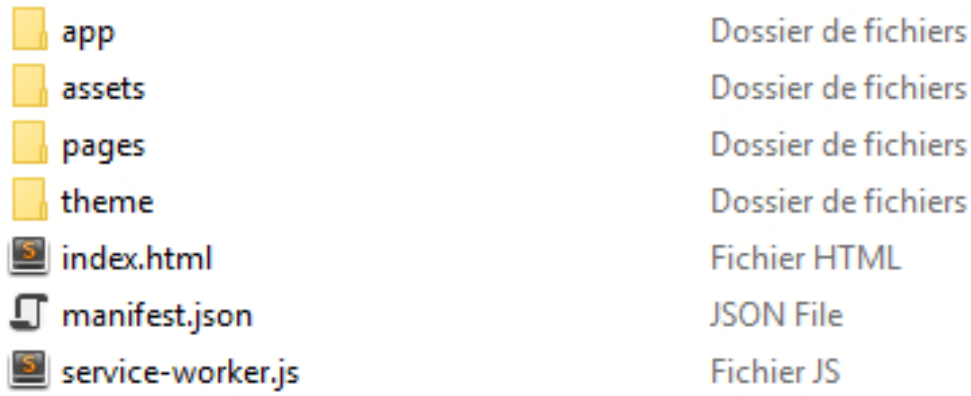
A savoir qu'on travaille uniquement dans le dossier « src », ce qui facilite les choses lors des premiers essais de développement, puisqu'on n'a pas besoin de connaître l'intégralité du projet pour pouvoir afficher le premier « Hello World » de l'application.

7.2 Dossier « src »

7.2.1 Contenu du dossier « src »

Le dossier qu'on va s'intéresser pour développer est le « src ». C'est lui qui va contenir les différents packages contenant les codes sources de l'application.

Figure 6 – Contenu du répertoire « src »



7.2.2 Contenu du dossier « app »

Dans le dossier « app », on y trouve différents fichiers et dossiers. Par exemple, dans le dossier « domaines », il y est stocké toutes les classes métiers du domaine. Dans le répertoire « providers », c'est là que sont stockées les classes permettant d'interagir avec la base de données, c'est en quelque sorte la partie backend de l'application. Toutes les requêtes vers la base de données SQLite sont codées ici. Dans le répertoire « settings », j'y ai stocké mes classes de constantes globales.

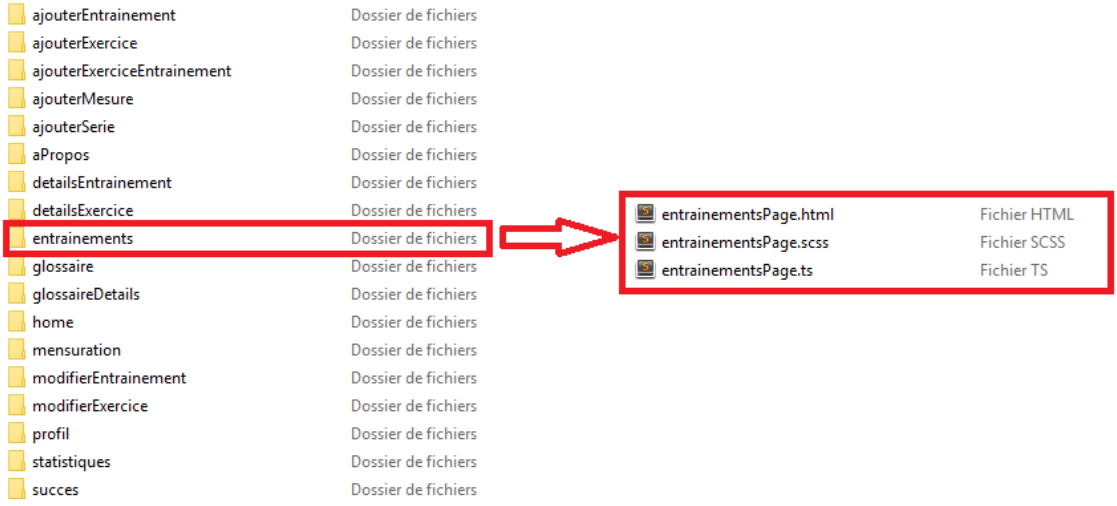
Ensuite, quelques fichiers utiles à l'application, comme le « app.components.ts » qui est l'instance de classe qui est créée en premier dans l'application et où il faut déclarer certaine dépendance. Après, le second fichier important est le « app.module.ts », c'est ici que sont déclarés et importés les modules utilisés dans l'application.

Figure 7 – Contenu du répertoire « app »

7.2.3 Contenu du dossier « pages »

Le dossier « pages » contient toutes les pages de l'application.

Figure 8 – contenu du répertoire « pages »



Chaque dossier comprend un fichier HTML, un SCSS et un TypeScript. Ce qui est pratique c'est qu'on se retrouve facilement dans le cas où on a mis des noms bien explicites. De plus, la maintenance d'un tel projet est simple à comprendre pour quelqu'un qui ne connait pas encore le projet.

7.2.4 Contenu du dossier « assets »

Le dossier « assets » contient simplement les fichiers images dont l'application à besoin. Par exemple, les images des exercices, des muscles, etc…

8. Ecrans principaux de l'application

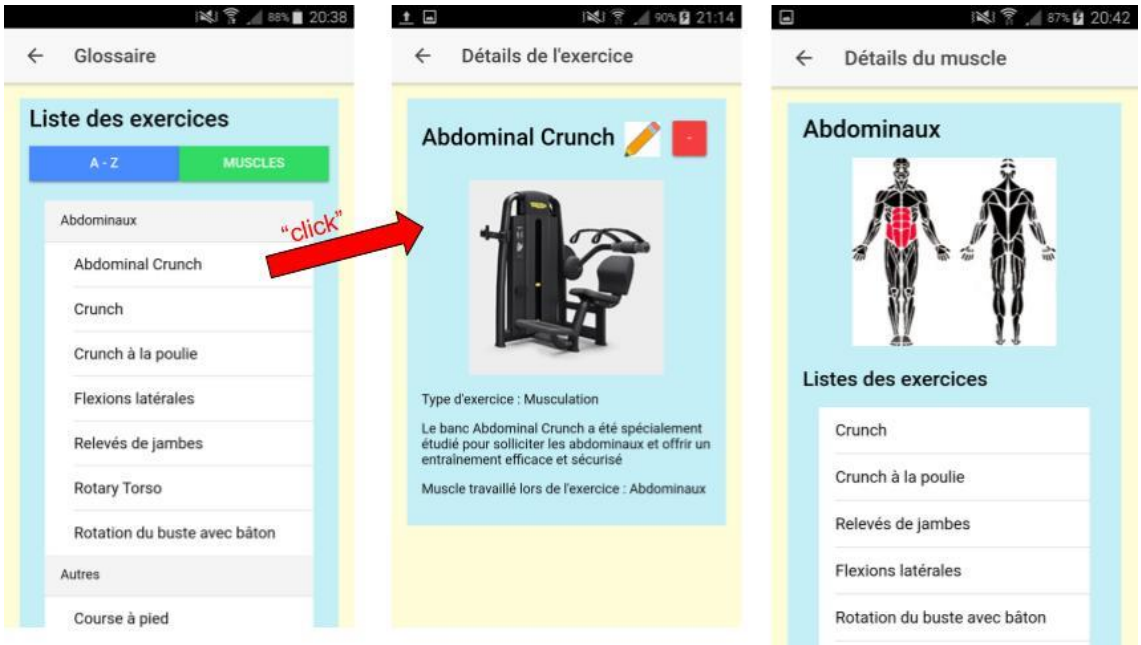
Tous les écrans suivants ont été réalisés par mes soins à l'aide des langages html et css et des composants disponibles dans la documentation du Framework Ionic.

8.1 Page d'accueil

Figure 9 – Capture d'écran de la page d'accueil de l'application

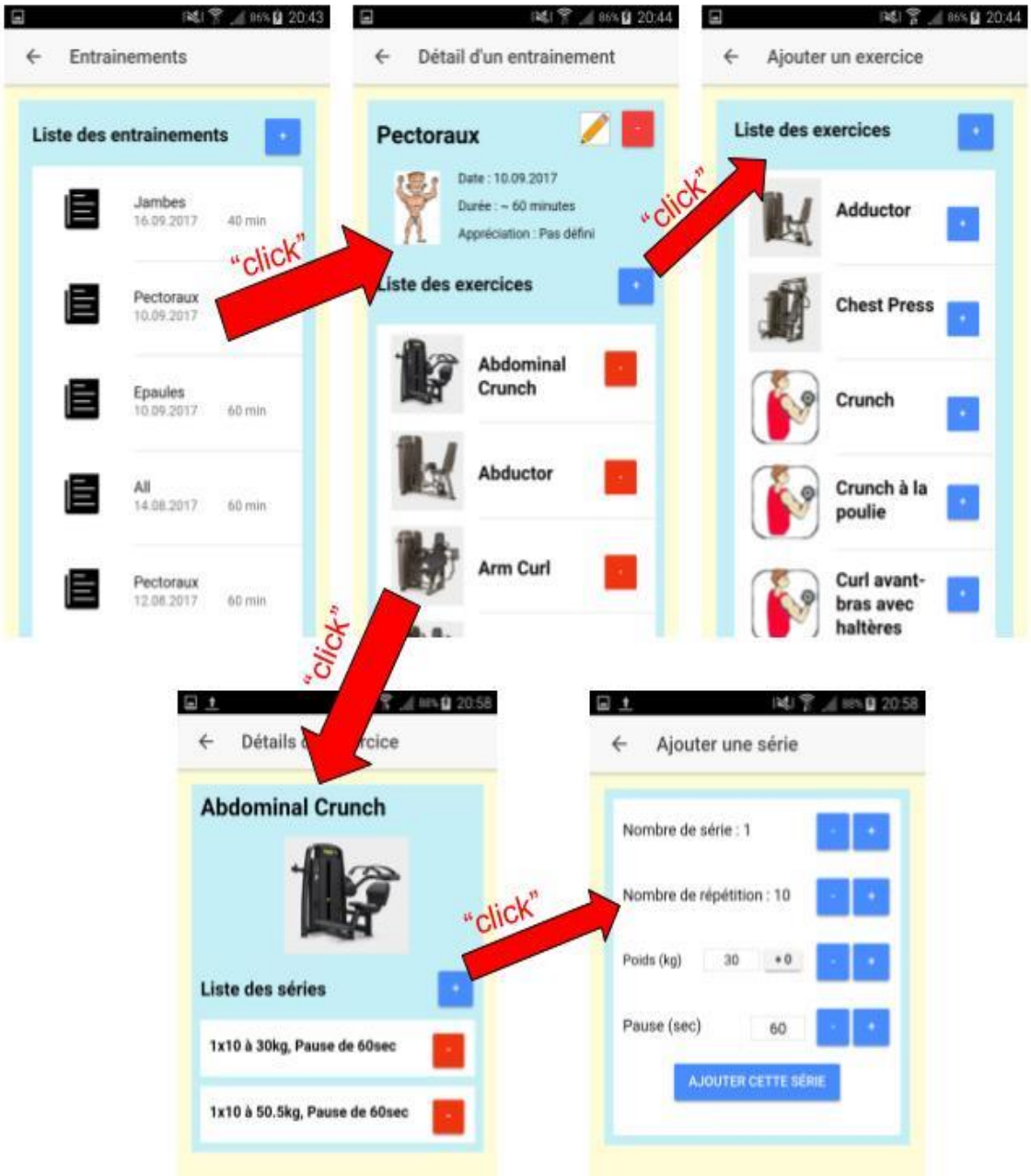
8.2 Glossaire

Figure 10 – Captures d'écrans de pages du glossaire



8.3 Gestion d’un entrainement

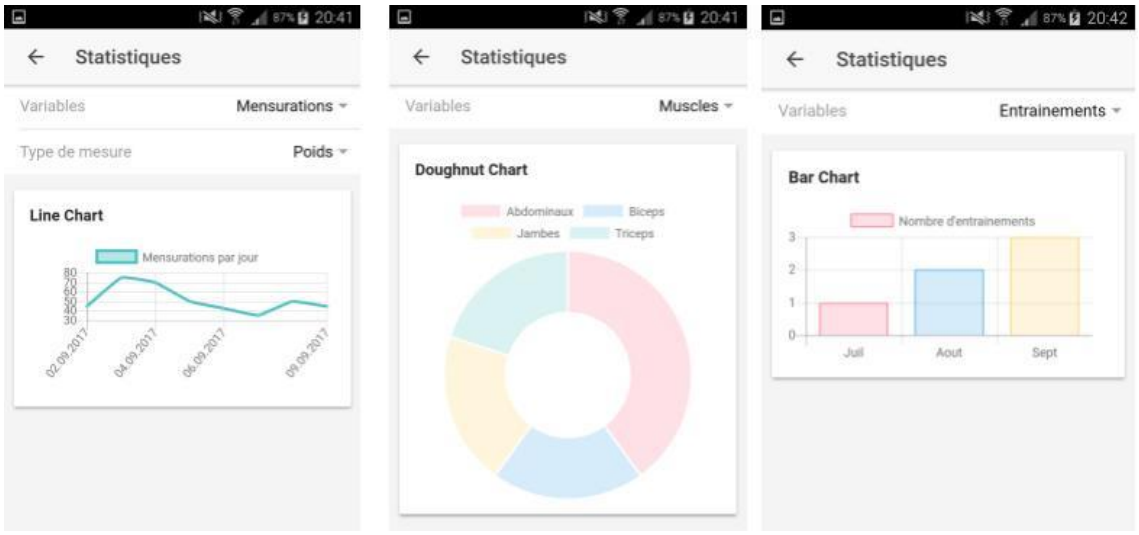
Figure 11– Captures d’écrans des pages de gestion d’entrainements



8.4 Statistiques

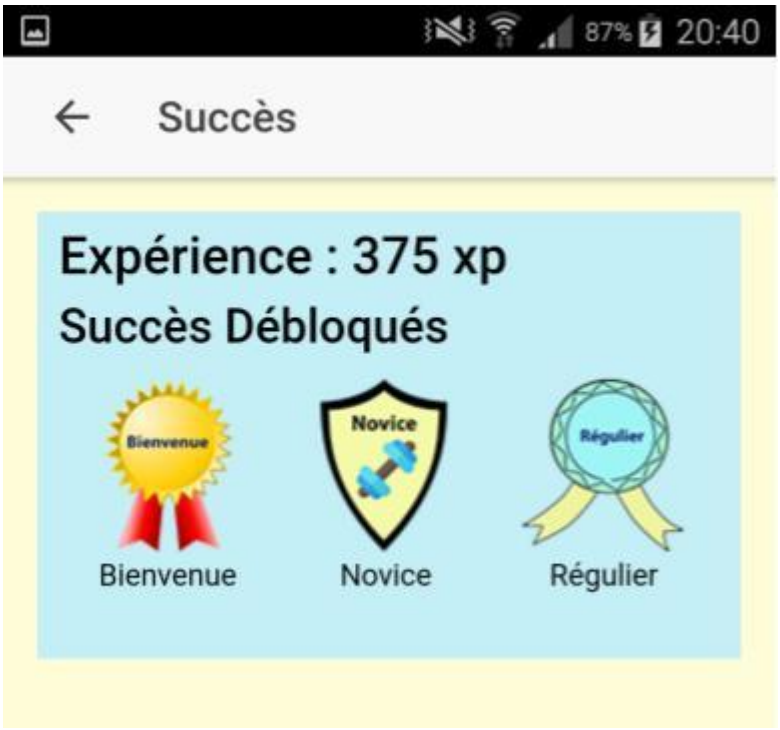
Les graphiques proviennent d'une librairie du nom de « Chart.js » que j'ai trouvé sur Internet, très simple d'utilisation et le rendu est impeccable. Il m'a fallu installer le package à l'aide de la commande « npm install chart.js –save » et le tour était joué. Il ne manquait plus qu'à importer la librairie dans la classe StatistiquesPage.

Figure 12 - Captures d'écrans de la page statistiques



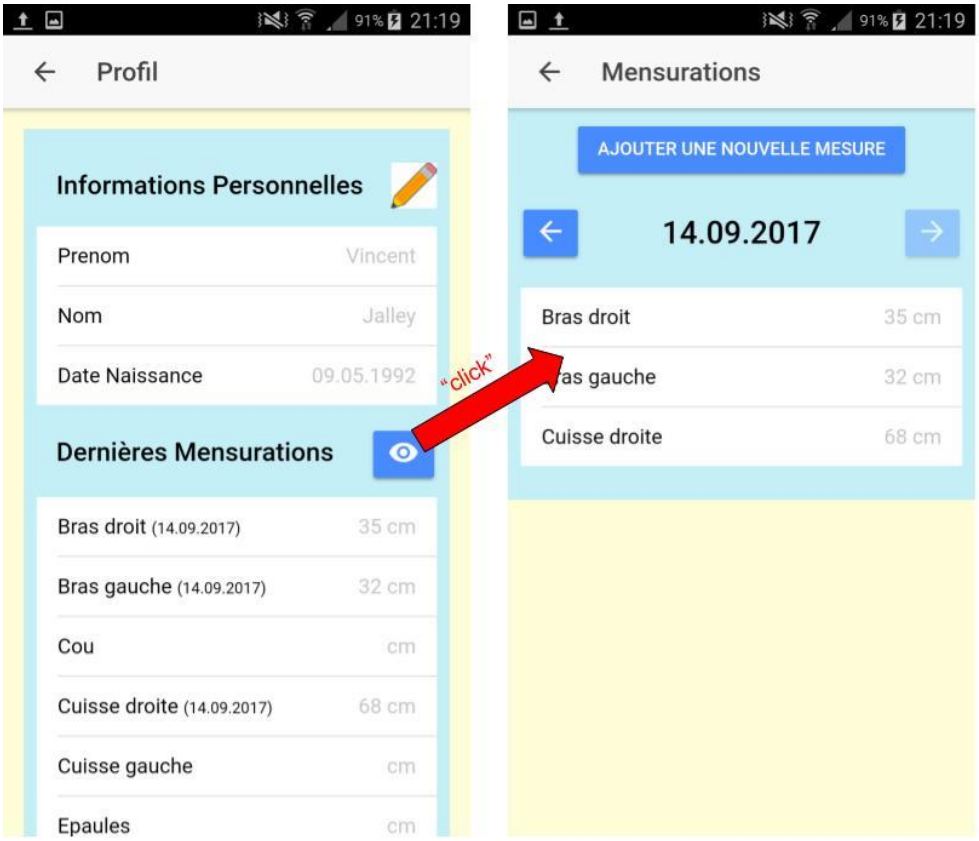
8.5 Succès

Figure 13 – Capture d'écran de la page succès



8.6 Profil

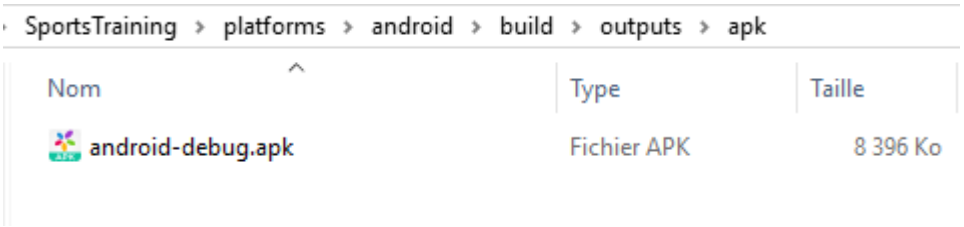
Figure 14 – Captures d'écrans de la page profil



9. Déploiement de l'application sous Android

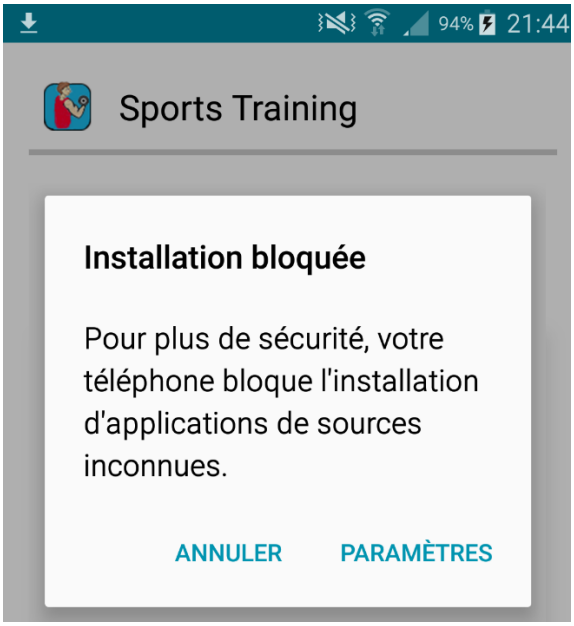
J'ai réalisé le déploiement uniquement sous Android, car je ne possède pas d'autre appareil. Le déploiement est assez simple une fois l'environnement installé. Il faut dans un premier temps, ajouter la plateforme Android au projet Ionic à l'aide de la commande « Ionic cordova add platform android ». Les commandes peuvent varier suivant les versions de Ionic, dans mon cas, j'utilise la dernière version de Ionic (version : 3.9). Ensuite, il suffit de construire l'exécutable devant être installé sur le smartphone à l'aide de la commande suivante « ionic cordova build android ». Après, il faut se rendre dans le dossier du projet et récupérer l'APK créée.

Figure 15 – Chemin d'accès de l'APK



L’application peut maintenant être installée sur un smartphone Android. Cependant, elle n’est pas prête à être uploader sur le Play Store de Google, car il faut passer par une étape supplémentaire pour valider l’application.

Figure 16 – Application Android de sources inconnues



S’il on essaye de l’installer sans « signer » l’application, Android préviendra qu’il s’agit d’une application de sources inconnues, mais l’on peut toujours l’installer, si l’on coche l’acceptation des installations de sources inconnues à l’écran suivant.

10. Exemple de quelques codes

10.1 Navigation entre les pages

La navigation entre les différentes pages se gère grâce à un composant du Framework Ionic. En effet, dans chaque classe, on a la possibilité d’importer avec l’injection de dépendance le composant « NavController » qui gère lui-même un stack permettant de faire un historique de navigation (pratique pour revenir en arrière) et d’afficher la dernière page ajoutée dans celui-ci. Par exemple, si on est sur la page « home » et qu’on veut se rendre sur la page « entrainements », il suffira d’une instruction dans le code pour ajouter une nouvelle page dans le stack du NavController.

Figure 17 – Exemples de code pour la gestion de la navigation

```
private afficherAjouterEnt () {
  this.navCtrl.push(AjouterEntrainementPage); // on ajoute au stack la page AjouterEntrainementPage
} // afficherAjouterEnt

private afficherBtnAnnuler() {
  this.navCtrl.pop (); // on enlève du stack la page courante
} // afficherBtnAnnuler
```

De plus, il y a la possibilité de passer des données à la vue suivante si on le désire.

10.2 Exemple d'une classe providers

Comme on peut le voir sur l'image ci-dessous, il y a plusieurs modules qui sont importés. Par exemple, NavController pour la navigation des pages, NavParams pour récupérer les données entre les navigations, SQLiteDAO afin d'en récupérer l'instance de la connexion à la base de données SQLite ou encore les différentes importations du domaine.

De plus, on annote la classe en Injectable, c'est-à-dire qu'on pourra l'utiliser ou l'on veut dans l'application. Il suffira de la rajouter dans le constructeur du composant et l'injection de dépense fera le travail et on pourra utiliser les méthodes de cette classe.

Figure 18 – Exemples de code pour une classe providers

```
import { Injectable } from '@angular/core';
import { NavController, NavParams } from 'ionic-angular';

// Native components
import { SQLite, SQLiteObject} from '@ionic-native/sqlite';
import { SqliteDAO } from '../../app/providers/sqliteDAO';

//domaines
import { Entrainement } from '../../app/domaines/entrainement';
import { Appreciation } from '../../app/domaines/appreciation';

@Injectable()
export class EntrainementDAO {

    constructor(public sqlite: SQLite, public sqliteDAO:SqliteDAO) {
    } // constructor

    public ajouterEntrainement(nom:string, date:string, duree:number, ) : void {
        //prepared statement pour les données à insérer !
        this.sqliteDAO.db.executeSql('INSERT INTO `spt_entrainement` (ent_nom, ent_date, ent_duree, ent_app_id,
            ent_uti_id) VALUES (?, ?, ?, ?, ?)', [ nom, date, duree, "1", "1" ])
        .then(() => {console.log("entrainement inserted");})
        .catch(e => console.log(e));
    } // ajouterEntrainement

    public updateEntrainement(id:number, nom:string, date:string, duree:number, idApp:number) : void {
        //prepared statement pour les données à insérer !
        this.sqliteDAO.db.executeSql('UPDATE `spt_entrainement` SET ent_nom = ?, ent_date = ?, ent_duree = ?,
            ent_app_id = ? WHERE ent_id = ?', [ nom, date, duree, idApp, id ])
        .then(() => {console.log("entrainement updated");})
        .catch(e => console.log(e));
    } // updateEntrainement

    public  getEntrainements () {
        return new Promise ((resolve, reject) => {
            let entrainements: Array<Entrainement> = [];
            let req = "SELECT ent_id, ent_nom, ent_date, ent_duree, app_id, app_nom FROM spt_entrainement JOIN
                spt_appreciation ON ent_app_id = app_id ORDER BY ent_date DESC";
            this.sqliteDAO.db.executeSql(req, {})
            .then((data) => {
                if(data == null) {reject(null);}
                if(data.rows) {
                    if(data.rows.length > 0) {
                        let ent : Entrainement;
```

10.3 Exemple d'une classe du domaine

Si on a l'habitude de la programmation orienté objet, une telle classe ne devrait pas effrayer un développeur. En effet, rien de bien particulier, comme on peut le voir, tous les attributs sont déclarés en privé, et les méthodes GET et SET sont publiques.

Le seul point qui m'a paru un peu bizarre au début, c'est que l'on ne peut pas construire plusieurs constructeurs avec des signatures différentes. C'est pour cela que j'ai opté pour un constructeur vide et qu'à chaque nouvelle instance, je mets à jours les données de l'objet avec les setters dont j'ai besoin.

Figure 19 – Exemples de code pour une classe du domaine

```
import { Appreciation } from './appreciation';

export class Entrainement {

  private id : number;
  private nom : string;
  private date : string;
  private duree : number;
  private appreciation : Appreciation;

  constructor() {} // constructor

  public getId () : number {return this.id;}
  public getNom () : string {return this.nom;}
  public getDate () : string {return this.date;}
  public getDuree () : number {return this.duree;}
  public getAppreciation () : Appreciation {return this.appreciation;}

  public setId (id:number) { this.id = id;}
  public setNom (nom:string) {this.nom = nom;}
  public setDate (date:string) {this.date = date;}
  public setduree (duree:number) {this.duree = duree;}
  public setAppreciation (appreciation:Appreciation) {this.appreciation = appreciation;}

} // Entrainement
```

10.4 Exemple d'une classe TypeScript

Comme on peut le voir, cela reste relativement simple à comprendre si l'on fait du développement JavaScript. On importe tout ce dont on a besoin et on annote la classe en Component. Tout ceci est en quelque sorte le Template réalisé par Ionic. On a juste besoin de le suivre correctement et tout se déroule bien.

Dans le constructeur, on voit qu'il y a plusieurs paramètres. L'injection de dépendance se déroule ici en particulier. Automatiquement, l'instance de la classe aura accès à l'instance des différents paramètres reçus dans le constructeur.

De plus, il y a la notion de promesse, c'est-à-dire des instructions asynchrones sont réalisées et on n'a pas besoin d'attendre le traitement de celui-ci pour continuer à travailler avec l'application. Au début, c'est assez déroutant car on ne sait pas vraiment comment traiter ce genre d'instruction. Beaucoup de bugs sont apparus avec des variables qui n'étaient pas initialisées, alors que les bonnes méthodes étaient appelées. Heureusement, une fois qu'on s'en aperçoit le problème est vite régler. Ici, il y a un exemple dans la méthode onCreate(), ou l'on met à jour une valeur dans le Local Storage et que ensuite, on exécute le code et pas avant que celui-ci soit terminé.

Figure 20 – Exemples de code pour une classe TypeScript d'Angular 2

```
import { Component } from '@angular/core';
import { NavController, NavParams } from 'ionic-angular';

//pages
import { DetailsEntrainementPage } from '../detailsEntrainement/detailsEntrainementPage';
import { AjouterEntrainementPage } from '../ajouterEntrainement/ajouterEntrainementPage';

//providers
import { EntrainementDAO } from '../../app/providers/entrainementDAO';

//ionic-native
import { NativeStorage } from '@ionic-native/native-storage';

//settings
import { Constantes } from '../../app/settings/constantes';//Constantes.NS_ID_ENTRAINEMENT

@Component({
  selector: 'entrainementsPage',
  templateUrl: 'entrainementsPage.html'
})
export class EntrainementsPage {

  entrainements;

  constructor(private navCtrl: NavController, private entrainementDAO: EntrainementDAO, private nativeStorage: NativeStorage) {
  } // constructor

  //methode déclanché quand on entrera dans la vue
  ionViewWillEnter() {this.onCreate ();} // ionViewWillEnter

  private onCreate () {
    this.entrainements = [];
    this.entrainementDAO.getEntrainements ().then((resolve) => {
      this.entrainements = resolve
    })
  } // onCreate

  private afficherDetailsEnt (id:number) {
    this.nativeStorage.setItem(Constantes.NS_ID_ENTRAINEMENT, {idEnt: id})
    .then(() => { this.navCtrl.push(DetailsEntrainementPage);}, error => console.error('Error storing item idEnt', error));
  } // afficherDetailsEnt

  private afficherAjouterEnt () {
    this.navCtrl.push(AjouterEntrainementPage); // on ajoute au stack la page AjouterEntrainementPage
  } // afficherAjouterEnt

  private afficherBtnAnnuler() {
    this.navCtrl.pop (); // on enlève du stack la page courante
  } // afficherBtnAnnuler

} // EntrainementsPage
```

11. Rapport de test

Tableau 1 – Rapport de test (sprint 2)

Test	Résultat Attendu	Résultat
Création du projet Ionic	Un Hello world est affiché dans l'application	OUI
Entrainement, Afficher les enregistrements	La liste des entrainements est affichée correctement	OUI
Entrainement, Ajouter un Entrainement	Le nouvel entrainement est bien ajouté dans la liste des entrainements	OUI
Entrainement, Afficher les détails d'un entrainement	Les informations d'un entrainement en particulier sont correctement affichées	OUI
Entrainement, Ajouter un exercice dans un entrainement	Après la sélection d'un exercice, il est ajouté à l'entrainement correspondant	OUI
Entrainement, Créer un exercice	La création d'un exercice affiche le nouvel exercice dans la liste des exercices	OUI
Entrainement, Ajouter une série à un exercice	La création d'une série est bien affichée dans l'exercice correspondant	OUI
Entrainement, Supprimer une série	Lorsque l'on clique sur le bouton de suppression, cela supprime bien en directe la série	OUI
Entrainement, Supprimer un exercice	Lorsque l'on clique sur le bouton de suppression d'exercice, cela supprime l'exercice directement de l'entrainement	OUI

Entrainement, Supprimer un entrainement	Lorsque l'on clique sur le bouton de suppression d'entrainement, cela supprime bien l'entrainement instantanément	OUI
Modifier un entrainement	Lorsque l'on modifie un entrainement, l'entrainement est bien mis à jour dans la fenêtre suivante.	OUI

Tableau 2 – Rapport de test (sprint 3)

Test	**Résultat Attendu**	**Résultat**
Glossaire, Afficher une liste d'exercice par ordre alphabétique	Une fois arrivé dans le glossaire, une liste d'exercice est affichée dans l'ordre alphabétique	OUI
Glossaire, Afficher une liste d'exercice triée par muscle	Lorsque l'on sélectionne le bouton « trier par muscle », la liste est mise à jour	OUI
Glossaire, Afficher une Liste de muscle	Lorsque l'on clique sur le bouton « muscle », cela affiche une liste de muscles triée par ordre alphabétique	OUI
Glossaire, afficher les détails d'un exercice	Lorsque l'on clique sur un exercice, les détails sont affichés	OUI
Glossaire, afficher les détails d'un muscle	Lorsque l'on clique sur un muscle, les détails sont affichés	OUI
Glossaire, modifier un exercice	Lorsque l'on modifie un exercice, l'exercice est bien mis à jour	OUI
Glossaire, supprimer un exercice	Lorsque l'on supprime un exercice, l'exercice n'existe plus dans l'application	OUI

Profil, afficher les informations de l'utilisateur	Quand on se rend sur la page du profil, les informations de l'utilisateur sont affichées	OUI
Profil, afficher les dernières mesures	Quand on se rend sur la page du profil, les dernières mesures sont affichées	OUI
Profil, Modifier les informations de l'utilisateur	Une fois que les informations de l'utilisateur sont modifiées, elles se mettent directement à jour	OUI
Profil, affiche le système de pagination des mesures	Lorsque l'on se rend dans la liste des mesures, les mesures sont affichées par jour	OUI
Profil, Ajouter une mesure	Une mesure est ajoutée directement à la liste des mesures et dans les dernières mesures enregistrées	OUI
Succès, Affiche l'expérience	Lorsque l'on se rend sur la page des succès, l'expérience de l'utilisateur est affichée	OUI
Succès, liste de succès	Les succès débloqués sont affichés dans la page	OUI

Tableau 3 – Rapport de test (sprint 4)

Test	**Résultat Attendu**	**Résultat**
Statistiques, afficher les statistiques des entrainements	Lorsque l'on se rend sur la page des statistiques, le graphique avec le nombre d'entrainements par mois est affiché par défaut	OUI

Statistiques, afficher les statistiques des exercices de musculation	Si l'on sélectionne la variable exercices de musculation, cela affiche une liste d'exercice et ensuite affiche un graphique correspondant	OUI
Statistiques, afficher les statistiques des exercices de cardio	Si l'on sélectionne la variable exercices de cardio, cela affiche une liste d'exercices et ensuite affiche un graphique correspondant	OUI
Statistiques, afficher les statistiques des mensurations	Si l'on sélectionne la variable mensurations, cela affiche une liste de mensurations et ensuite affiche un graphique correspondant	OUI
Statistiques, afficher les statistiques des muscles	Si l'on sélectionne la variable muscles, cela affiche un graphique en lien	OUI
A propos, Afficher les informations	Si l'on se rend sur la page à propos, cela affiche bien les informations de l'application	OUI

12. Conclusion

Les technologies évoluent et l'on est sans cesse obligé d'en apprendre des nouvelles pour rester à jour. C'est ce qui m'a motivé à l'accomplissement de ce projet avec le Framework Ionic. En effet, je ne connaissais au début ce Framework que de nom, mais maintenant je peux me débrouiller et créer des applications mobiles hybrides. Avant ce travail de Bachelor, je n'avais jamais entendu parler de ce genre d'application où l'on pouvait avoir un seul code source pour plusieurs plateformes cibles. J'ai trouvé le concept génial, parce que le code était mélangé de plusieurs Framework et langages différents.

J'ai eu un module à la HEG dans lequel on devait développer des applications mobiles pour Android en Java. Je ne saurais expliquer pourquoi, mais je n'ai pas vraiment apprécié développer dans ce langage. Alors qu'avec le Framework Ionic, il y avait vraiment tout ce dont j'aimais dans la programmation, une partie design à réfléchir et concevoir de mes mains avec les langages HTML et CSS, une autre avec la logique en JavaScript dans les classes TypeScript d'Angular 2 et surtout du développement d'application web.

De plus, je me suis vraiment senti libre dans ce projet, car j'ai apporté les fonctionnalités que j'avais besoin pour mon application de fitness. Aujourd'hui, j'ai réalisé l'application dont j'avais l'envie depuis des années, mais pas vraiment le temps et la motivation de les réaliser. Le travail de Bachelor est très bien tombé et je ne regrette pas du tout mon choix de projet.

Des difficultés, j'en ai rencontré tous les jours, car apprendre un Framework de soit même est bien plus compliqué qu'apprendre le Java à la HEG avec un professeur qui nous explique le langage. J'ai dû suivre les bonnes pratiques qu'on m'a enseigné, c'est-à-dire lire les fameuses documentations écrites par le Framework. Heureusement, La documentation en ligne était très bien faite et cela m'a beaucoup aidé dans la compréhension du Framework et la réalisation de l'application.

Un souvenir me revient, c'était il y a quelques années quand j'ai dû réaliser un site web pour mon travail de fin de CFC d'informaticien. Les seuls langages que j'avais utilisés étaient du PHP, HTML et CSS. De plus, quand j'ai relu ce code, il y a quelques mois, c'était incompréhensible et impossible de réaliser une maintenance dessus. Aujourd'hui avec ce projet de Bachelor, je pense vraiment avoir développé un projet simple,

compréhensible et surtout maintenable, car j'ai respecté les bonnes pratiques de programmation apprises à la HEG.

Ce travail est le dernier que je ferais à la Haute Ecole de Gestion de Genève et je pense avoir fièrement montré tout ce dont je suis capable, ainsi qu'une bonne partie des compétences apprises durant ces trois années d'école.

Bibliographie

Sites web

IONIC FRAMEWORK, 2017, documentations de Ionic, [consulté le 15.09.2017] Disponible à l'adresse : http://ionicframework.com/docs/

JOSHMORONY, 2017, Adding Responsive Charts & Graphs to Ionic 2 & 3 Applications, [consulté le 02.09.2017], Disponible à l'adresse : https://www.joshmorony.com/adding-responsive-charts-graphs-to-ionic-2-applications/

TECHNOGYM, 2017, produits, [consulté le 20.08.2017], Disponible à l'adresse : https://www.technogym.com/ch/fr/products.html

AKELYS, 2017, Informations sur les exercices de musculation, [Consulté le 09.09.2017], Disponible à l'adresse : http://www.akelys.com/exercices/exercices_par_muscle/muscles.html

MDN WEB DOCS, 2017, JavaScript, [Consulté le 20.07.2017], Disponible à l'adresse : https://developer.mozilla.org/en-US/docs/Web/JavaScript

ESPACE-MUSCULATION, 2017, Exercice et mouvements de musculation, [Consulté le 03.08.2017], Disponible à l'adresse : https://www.espace-musculation.com/exercices/

CORDOVA, 2017, Android Platform Guide, [Consulté le 05.07.2017], Disponible à l'adresse : https://cordova.apache.org/docs/en/7.x/guide/platforms/android/

ANGULAR 2, 2017, What is Angular, [Consulté le 05.07.2017], Disponible à l'adresse : https://angular.io/docs

Blogs

Weblog, 2015, Ionic une puissante alternative hybride au développement natif d'application mobile, [consulté le 02.09.2017] Disponible à l'adresse : http://blog.wemanity.com/ionic-une-puissante-alternative-hybride-au-developpement-natif-dapplication-mobile/

Vidéos

YOUTUBE, 6 aout 2015, Tutoriel : Créer une application Ionic Framework par Graphikart, [Consulté le 03.07.2017] Disponible à l'adresse :

https://www.youtube.com/watch?v=2GCeWf75nN4

YOUTUBE, 2017, Développement Ionic 3 par TheiPhoneRetro, [Consulté le 05.07.2017], Disponible à l'adresse : https://www.youtube.com/watch?v=7pwgdSCS5g0&list=PL8Azg5184hTC9lgukBd_okwj5mgx2mpHK

Annexe 1: SCRUM - Backlog

ID Storie	Thème	En tant que...	Je veux...	Afin de....	Priorité	Difficulté	Heures Estimées	Etat
A	Recherche	Développeur	rechercher de la documentation	pouvoir commencer le TB dans de bonne condition	1	8	16	Terminé
B	Formation	Développeur	connaître les spécifications du Framework Ionic	faciliter la programmation	2	13	8	Terminé
C	Formation	Développeur	connaître les spécifications du Framework Angular 2	faciliter la programmation	3	20	8	Terminé
D	BDD	Concepteur	réaliser les maquettes de l'application	réaliser la base de données	4	5	24	Terminé
E	BDD	Concepteur	concevoir les modèles de données	créer les scripts de la base de données	5	13	24	Terminé
F	Programmation	Développeur	créer le projet Ionic	voir que tout fonctionne	6	3	4	Terminé
G	Programmation	utilisateur	réaliser la fonctionnalité de gestion des entrainements	d'enregistrer des entrainements de fitness	7	20	80	Terminé
H	Programmation	Développeur	réaliser le glossaire de l'application	pouvoir rechercher un exercice ou muscle particulier	8	13	24	Terminé
I	Programmation	Développeur	réaliser le profil de l'utilisateur	pouvoir enregistrer des mesures	9	13	24	Terminé
J	Programmation	Développeur	réaliser les succès de l'application	motiver l'utilisateur à rester sur l'application	10	8	16	Terminé
K	Programmation	Développeur	réaliser les statistiques des entrainements	pouvoir afficher l'évolution d'une variable	11	20	32	Terminé
L	Programmation	Développeur	réaliser la page A Propos de l'application	renseigner l'auteur de l'application	12	3	1	Terminé
M	Rédaction	Etudiant	Rédiger le document de l'application	réussir le travail de Bachelor	13	13	80	Terminé
						Temps restant :	**-1**	

Annexe 2: SCRUM - Sprint Backlog (1/2)

Sprint ID	Story ID	Noms de la tâche	Etat	Début	Fin	Prédécesseurs	Noms ressources	% achevé	Estimation en heure	Total estimation
1	A			lun 03.07.17	ven 14.07.17	-				72
		Faire de la recherche sur les technologies mobiles	Terminé				V. Jalley	100%	8	
	B			lun 03.07.17	ven 14.07.17	-				
		Lire la documentation du Framework Ionic	Terminé				V. Jalley	100%	2	
		Rechercher des tutoriels vidéos	Terminé				V. Jalley	100%	2	
		Comprendre le fonctionnement du Framework	Terminé				V. Jalley	100%	4	
	C			lun 03.07.17	ven 14.07.17	-				
		Lire la documentation du Framework Angular 2	Terminé				V. Jalley	100%	2	
		Rechercher des tutoriels vidéos	Terminé				V. Jalley	100%	2	
		Comprendre le fonctionnement du Framework	Terminé				V. Jalley	100%	4	
	D		Terminé	lun 03.07.17	ven 14.07.17					
		Dessiner les maquettes des différentes fonctionnalités	Terminé				V. Jalley	100%	24	
	E		Terminé	lun 03.07.17	ven 14.07.17					
		Concevoir un premier jet de la base de données	Terminé				V. Jalley	100%	24	
2	F			lun 17.07.17	mer 16.08.17	A, B, C, D, E				112
		Créer la première version du projet Ionic	Terminé				V. Jalley	100%	2	
	G			lun 17.07.17	mer 16.08.17	A, B, C, D, E				
		Réaliser les interfaces de gestion d'entrainement	Terminé				V. Jalley	100%	40	
		Créer la base de données SQLite	Terminé				V. Jalley	100%	8	
		Créer les classes du domaines	Terminé				V. Jalley	100%	1	
		Afficher des entrainements	Terminé				V. Jalley	100%	3	
		Ajouter un entrainement	Terminé				V. Jalley	100%	5	
		Afficher les détails d'un entrainement	Terminé				V. Jalley	100%	4	
		Ajouter un exercice à l'entrainement	Terminé				V. Jalley	100%	8	
		Créer un nouvel exercice	Terminé				V. Jalley	100%	12	
		Ajoute une série	Terminé				V. Jalley	100%	16	
		Supprimer une série, un exercice, un entrainement	Terminé				V. Jalley	100%	8	
		Modifier un entrainement	Terminé				V. Jalley	100%	5	

Annexe 3: SCRUM - Sprint Backlog (2/2)

Sprint ID	Story ID	Noms de la tâche	Etat	Début	Fin	Prédécesseurs	Noms ressources	% achevé	Estimation en heure	Total estimation
	H			jeu 17.08.17	ven 25.08.17	F, G				
		Afficher une liste d'exercices triée par ordre alphabétique	Terminé				V. Jalley	100%	2	
		Afficher une liste d'exercices triée par muscle	Terminé				V. Jalley	100%	4	
		Afficher une liste de muscles triée par ordrre alphabétique	Terminé				V. Jalley	100%	2	
		Afficher les détails d'un exercice	Terminé				V. Jalley	100%	4	
		Afficher les détails d'un muscle	Terminé				V. Jalley	100%	2	
		Modifier un exercice	Terminé				V. Jalley	100%	3	
		Supprimer un exercice	Terminé				V. Jalley	100%	4	
	I			jeu 17.08.17	ven 25.08.17	F, G				
3		Afficher les informations de l'utilisateur	Terminé				V. Jalley	100%	6	**71**
		Modifier les informations de l'utilisateur	Terminé				V. Jalley	100%	6	
		Afficher les dernières mesure effectuées	Terminé				V. Jalley	100%	4	
		Afficher le système de pagination des mesures enregistrées	Terminé				V. Jalley	100%	8	
		Ajouter une mesure	Terminé				V. Jalley	100%	8	
	J			jeu 17.08.17	ven 25.08.17	F, G				
		Afficher l'expérience de l'utilisateur	Terminé				V. Jalley	100%	2	
		Créer les images des succès	Terminé				V. Jalley	100%	4	
		Afficher les succès débloqués	Terminé				V. Jalley	100%	4	
		Afficher message des succès débloqués	Terminé				V. Jalley	100%	8	
	K			ven 01.09.17	ven 15.09.17	H, I, J				
		Rechercher librairie pour afficher des graphiques	Terminé				V. Jalley	100%	2	
		Afficher des statistiques sur les entrainements	Terminé				V. Jalley	100%	6	
		Afficher des statistiques sur les exercices de musculation	Terminé				V. Jalley	100%	2	
		Afficher des statistiques sur les exercices de cardio	Terminé				V. Jalley	100%	2	
4		Afficher des statistiques sur les mensurations	Terminé				V. Jalley	100%	2	**88**
		Afficher des statistiques sur les muscles	Terminé				V. Jalley	100%	2	
	L			ven 01.09.17	ven 15.09.17	H, I, J				
		Afficher les informations de l'application	Terminé				V. Jalley	100%	2	
	M			ven 01.09.17	ven 15.09.17	H, I, J				
		Réaliser la documentation du Travail de bachelor	Terminé				V. Jalley	100%	70	

Annexe 4 : SCRUM - Sprint Review (1/4)

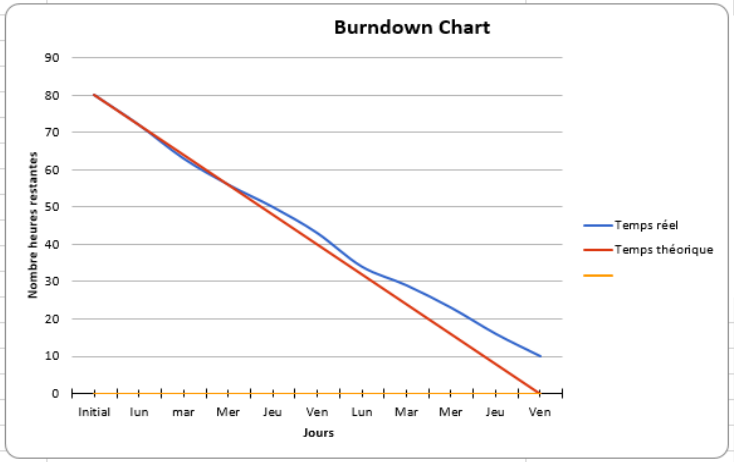
Sprint 1

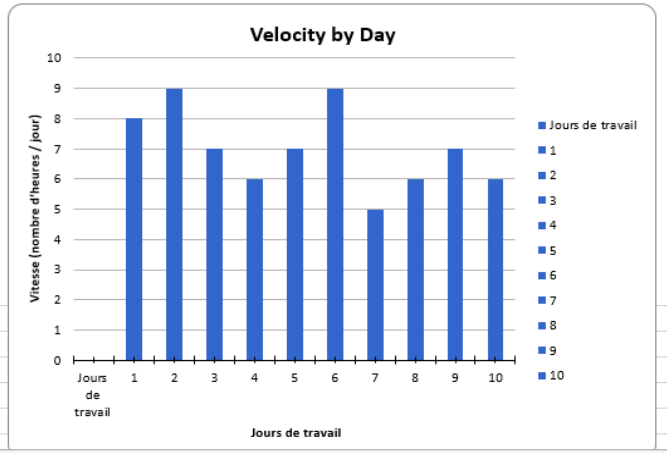
Date début	lundi 3 juillet 2017
Date fin	vendredi 14 juillet 2017
Nb heures	80

1 jour = 8 h

Objectif du sprint	ID Story
rechercher de la documentation	**A**
connaître les spécifications du Framework Ionic	**B**
connaître les spécifications du Framework Angular 2	**C**

ID	Short name	Value/Importance	10	lun	mar	Mer	Jeu	Ven	Lun	Mar	Mer	Jeu	Ven
A	Faire de la recherche sur les technologies mobiles	1	8	**8**									
B	Lire la documentation du Framework Ionic	2	2		**2**								
	Rechercher des tutoriels vidéos	3	5		**5**								
	Comprendre le fonctionnement du Framework	4	2		**2**								
C	Lire la documentation du Framework Angular 2	5	2			**2**							
	Rechercher des tutoriels vidéos	6	4			**4**							
	Comprendre le fonctionnement du Framework	7	1			**1**							
D	Dessiner les maquetes des différentes fonctionnalités	8	27				**6**	**7**	**9**	**5**			
E	Concevoir un premier jet de la base de données	9	19								**6**	**7**	**6**
			70	8	9	7	6	7	9	5	6	7	6





Annexe 5 : SCRUM - Sprint Review (2/4)

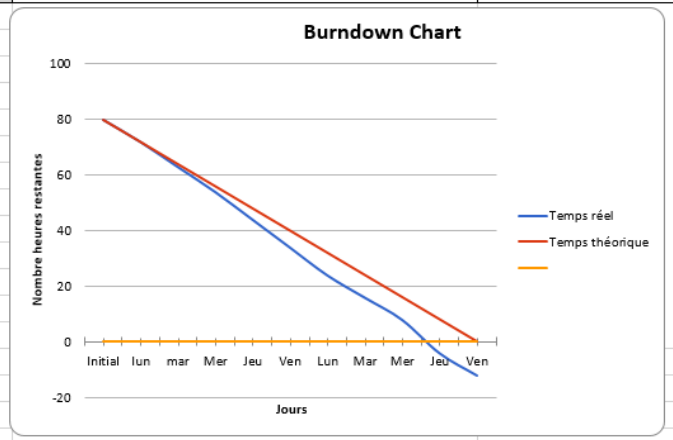
Sprint 2

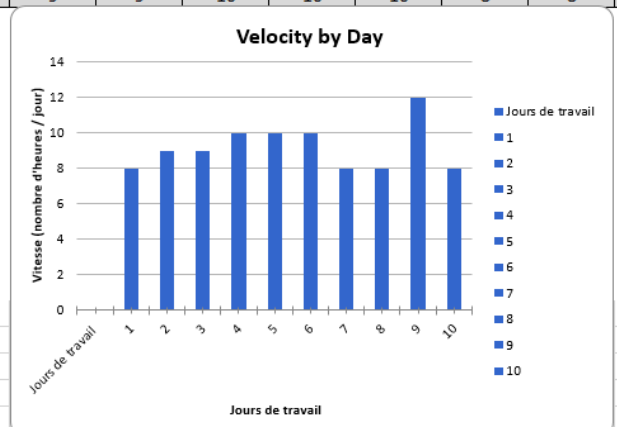
Date début	lundi 17 juillet 2017
Date fin	mercredi 16 août 2017
Nb heures	80

1 jour = 8 h

Objectif du sprint	ID Story
créer le projet Ionic	**F**
réaliser la fonctionnalité de gestion des entrainements	**G**

ID	Short name	Value/Importance	-12	lun	mar	Mer	Jeu	Ven	Lun	Mar	Mer	Jeu	Ven
F	Créer la première version du projet Ionic	1	2	**2**									
	Réaliser les interfaces de gestion d'entrainement	2	34	**6**	**9**	**9**	**10**						
	Créer la base de données SQLite	3	10					**10**					
	Créer les classes du domaines	4	3						**3**				
	Afficher des entrainements	5	3						**3**				
	Ajouter un entrainement	6	4						**4**				
G	Afficher les détails d'un entrainement	7	4							**4**			
	Ajouter un exercice à l'entrainement	8	8							**4**	**4**		
	Créer un nouvel exercice	9	4								**4**		
	Ajoute une série	10	12									**12**	
	Supprimer une série, un exercice, un entrainement	11	4										**4**
	Modifier un entrainement	12	4										**4**
			92	8	9	9	10	10	10	8	8	12	8





Annexe 6 : SCRUM - Sprint Review (3/4)

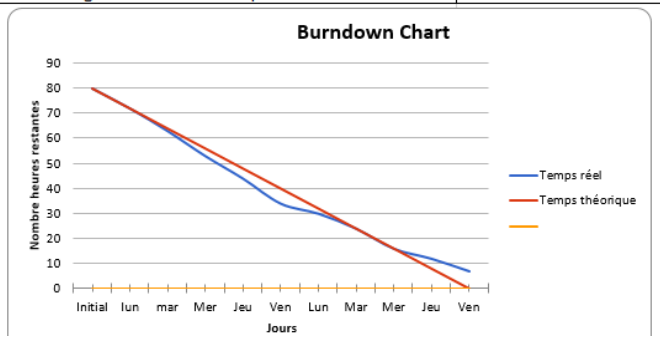
Sprint 3

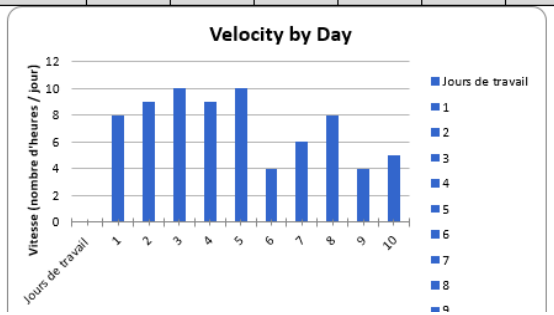
Date début	jeudi 17 août 2017
Date fin	vendredi 25 août 2017
Nb heures	80

1 jour = 8 h

Objectif du sprint	ID Story
réaliser le glossaire de l'application	**H**
réaliser le profil de l'utilisateur	**I**
réaliser les succès de l'application	**J**

ID	Short name	Value/Importance	7	lun	mar	Mer	Jeu	Ven	Lun	Mar	Mer	Jeu	Ven
H	Afficher une liste d'exercices triées par ordre alphabétique	1	2	**2**									
	Afficher une liste d'exercices triées par muscle	2	4	**4**									
	Afficher une liste de muscles triées par ordrre alphabétique	3	2	**2**									
	Afficher les détails d'un exercice	4	4		**4**								
	Afficher les détails d'un muscle	5	2		**2**								
	Modifier un exercice	6	3		**3**								
	Supprimer un exercice	7	4			**4**							
I	Afficher les informations de l'utilisateur	8	6			**6**							
	Modifier les informations de l'utilisateur	9	4				**4**						
	Afficher les dernières mesure effectuées	10	4				**4**						
	Afficher le système de pagination des mesures enregistrées	11	9				**1**	**8**					
	Ajouter une mesure	12	6					**2**	**4**				
J	Afficher l'expérience de l'utilisateur	13	2							**2**			
	Créer les images des succès	14	4							**4**			
	Afficher les succès débloqués	15	9								**4**		**5**
	Afficher message des succès débloqués	16	8								**4**	**4**	
			73	8	9	10	9	10	4	6	8	4	5





Annexe 7 : SCRUM - Sprint Review (4/4)

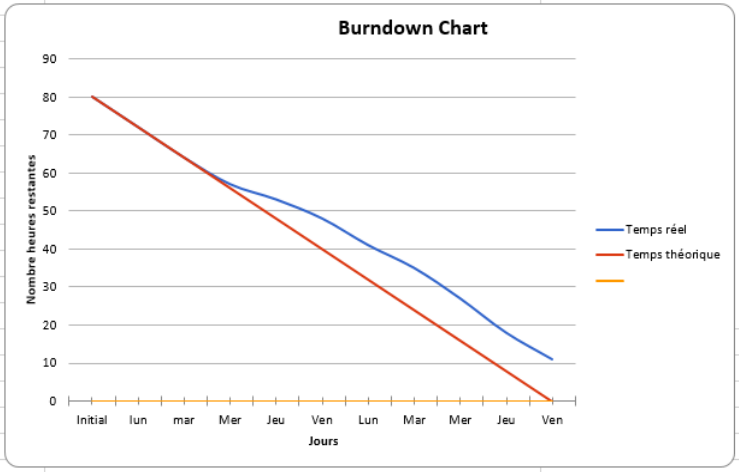
Sprint 4

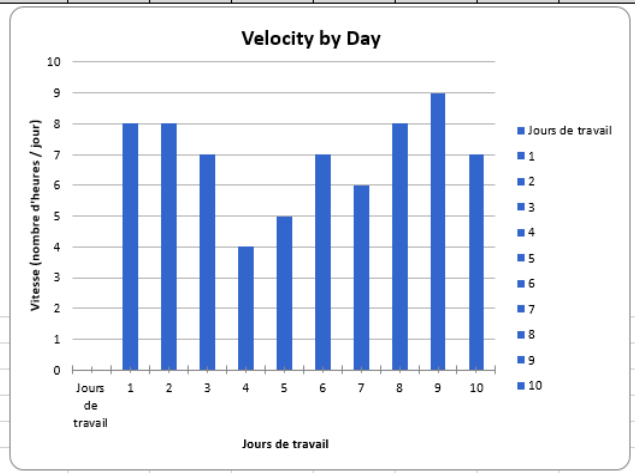
Date début	vendredi 1 septembre 2017
Date fin	vendredi 15 septembre 2017
Nb heures	80

1 jour = 8 h

Objectif du sprint	ID Story
réaliser les statistiques des entrainements	**K**
réaliser la page A Propos de l'application	**L**
Rédiger le document de l'application	**M**

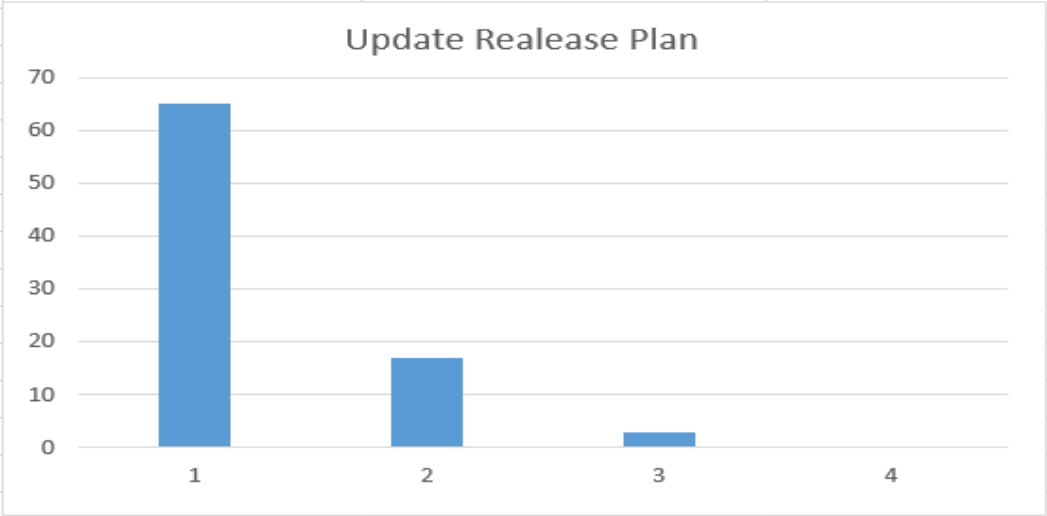
ID	Short name	Value/Importance	11	lun	mar	Mer	Jeu	Ven	Lun	Mar	Mer	Jeu	Ven
K	Rechercher librairie pour afficher des graphiques	1	2	**2**									
	Afficher des statistiques sur les entrainements	2	6	**6**									
	Afficher des statistiques sur les exercices de musculation	3	2		**2**								
	Afficher des statistiques sur les exercices de cardio	4	2		**2**								
	Afficher des statistiques sur les mensurations	5	2		**2**								
	Afficher des statistiques sur les muscles	6	2		**2**								
L	Afficher les informations de l'application	7	2			**2**							
M	Réaliser la documentation du Travail de bachelor	8	51			**5**	**4**	**5**	**7**	**6**	**8**	**9**	**7**
			69	8	8	7	4	5	7	6	8	9	7





Annexe 8 : SCRUM – Release Burndown Chart

formule :	resultat = (release workload / average iteration velocity) * iteration length			
Sprint	resultat	release workload	average iteration velocity	iteration length
1	65,14285714	190	8,75	3
2	17,04347826	98	11,5	2
3	2,739726027	25	9,125	1
4	0	-44	8,625	0



Annexe 9 : SCRUM - Diagramme de Gantt

Le projet dure 340 heures de travail sur une durée de 8 semaines à plein-temps